REMARKS

The Examiner is thanked for the performance of a thorough search and for reconsidering the rejections made in the Final Office Action mailed October 30, 2007 ("Office Action") in light of Applicant's reply filed December 27, 2007. By this amendment, Claims 1-2, 4-6, 17-18, 20, 27-31, 49, and 50 are amended. No claims are canceled, added, or withdrawn. Therefore, Claims 1-2, 4-10 and 12-50 are pending in the application.

CLAIMS REJECTIONS – 35 U.S.C. § 102

The pending claims stand rejected under 35 U.S.C. § 102(e) as allegedly being anticipated by US 2007/0233871 ("*Fletcher*"). This rejection is respectfully traversed.

AMENDMENTS TO THE CLAIMS ARE FULLY SUPPORTED IN THE SPECIFICATION

Claims 1, 17, 49 and 50 are amended to recite, in part:

the request having first data that cannot be used by said particular web service to service requests for said information

Support for this amendment can be found throughout the specification including at least in paragraphs 26 and 27. Paragraphs 26 and 27 describes embodiments of the present invention where data in a request from a client application is transformed to the proper data and format requirements of a particular web service, and passed to the particular web service according to access style and encoding requirements of the particular web service. One skilled in the art, in light of the description in paragraphs 26 and 27, would reasonably conclude then that the request from the client application may have "first data that cannot be used by said particular web service to service requests for said information".

Claims 1, 17, 49 and 50 are amended to recite, in part:

wherein the particular web service serves as a source of said information [and] is separate from the web services broker

Support for this amendment can be found throughout the specification including at least in paragraph 3 (providing one definition of web service as "any information source running

business logic processes conveniently packaged for use by an application or end-user.") and FIG. 1 (showing Web Service 106a-n separated by Network 109 from Web Service Broker 104). One skilled in the art, in light of the specification, would reasonably conclude then that the claimed "particular web service" could serve "as a source of said information" and be "separate from the web services broker".

Therefore, no new matter is added by the amendments herein and the amendments are fully supported in the specification.

FLETCHER'S PORTLET PROXY IS NOT THE CLAIMED "PARTICULAR WEB SERVICE"

As a preliminary matter, it should be noted that the portlet proxies of *Fletcher* differ from the particular web service of Claim 1 at a fundamental level. Specifically, Claim 1 recites "receiving, from a client application, a request for information from a particular web service that serves as a source of said information". Thus, to qualify as the particular web service of Claim 1 it must be a thing that is an information **source** for the requested information. In contrast, in *Fletcher*, a portlet proxy acts "as an intermediary between an application or software resource requesting a particular service and a software resource providing that service." (*Fletcher*, paragraph 46). Clearly, such an intermediary is not an information **source**. Thus, at very fundamental level, *Fletcher*'s portlet proxy cannot possibly be equated with the claimed "particular web service" of Claim 1.

At best, *Fletcher*'s software resource may be equated with the claimed particular web service. However, as explained below, *Fletcher*'s portlet proxies do not broker requests between client applications and software resources in the manner that the claimed web services broker of Claim 1 brokers requests between client applications and web services.

DISTINCTION BETWEEN WEB SERVICES BROKER AND WEB SERVICES PROXY

Claim 1 has specific limitations related to a "web services broker" while *Fletcher* discloses a web services proxy. (*See Fletcher*, paragraphs 46-47). It is critical for proper understanding of *Fletcher* these two concepts are not confused.

Claim 1 makes clear that the claimed "web services broker" receives, from a client application:

> a request for information from a particular web service, the request having first data that cannot be used by said particular web service to service requests for said information

Thus, the web services broker of Claim 1 is able to broker requests between client applications that request information from a particular web service and the particular web service that provides the requested information. Specifically, the claimed web services broker is able to broker requests that are not compatible with the input data requirements of the particular web service. The claimed web services broker does the brokering by "accessing transformation information that specifies how to transform said first data associated with said request to second data that said particular web service can use to service requests for said requested information" and then "based on said transformation information, … transforming said first data to said second data".

In contrast, *Fletcher*'s portlet proxy merely forwards request messages from client applications to the software resource to which the portlet proxy is bound. (See *Fletcher*, paragraph 46 stating "In either case (i.e., statically bound software resource or dynamically bound software resource), the portlet proxy receives request messages and forwards them to the software resource to which it is bound"). Clearly then, since the portlet proxy of *Fletcher* is simply forwarding request messages from client applications onto the software resource bound to the proxy, the request messages disclosed in *Fletcher* must be compatible with the input data

requirements of the software resource. Thus it would not make any sense for *Fletcher*'s portlet proxy to receive from a client application:

> a request for information from a particular web service, **the request having first data that cannot be used by said particular web service to service requests for said information**

Further, since *Fletcher*'s portlet proxy merely forwards a request from a client application onto a software resource, the client application in *Fletcher* must necessarily have logic for directly interacting with the software resource. In other words, given that the client application in *Fletcher* has logic for directly interacting with a portlet proxy and the portlet proxy forwards requests from client applications onto the software resource bound to the portlet proxy, the client application must necessarily have logic for directly interacting with the software resource. Thus it also does not make any sense to equate *Fletcher*'s client application with "a client application, separate from the web services broker, that does not have logic for directly interacting with said particular web service".

Additionally, since *Fletcher*'s portlet proxy simply performs a forwarding function when it receives a request for information from a software resource, the portlet proxy does not access transformation information that specifies how to transform data associated with the request to data that the software resource can use to provide the requested information. Thus, *Fletcher*'s portlet proxy does not teach or suggest the following limitations of Claim 1, which relate to how the claimed "web services broker" brokers requests from client applications:

> **in response to receiving said request**, the web services broker **accessing transformation information** that specifies
> how to transform said first data associated with said request to second data that said particular web service can use to service requests for said requested information,
> how to invoke said particular web service in a manner required by said particular web service, to obtain said requested information from said particular web service, and

how to transform a plurality of first data each from a respective client application of a plurality of client applications, to a plurality of second data each for a respective web service of a plurality of web services;

based on said transformation information, the web services broker performing the steps of:

transforming said first data to said second data; and

invoking, in said manner required by said particular web service, said particular web service to obtain said requested information from said particular web service.

ADVANTAGES OFFERED BY THE CLAIMED WEB SERVICES BROKER

One advantage offered by the claimed web services broker is that client applications may be developed without knowledge of the input data specification, service invocation style, and/or service invocation format associated with a particular web service. (*See Specification*, paragraph 54). Specifically, the claimed transformation information can be created and supplied to the web services broker after the client application has been developed and deployed. (*See Specification*, paragraph 54 discussing application data-capturing and profile-capturing activities). This allows client applications to developed more generically, without being written specifically to invoke the particular web service.

In contrast, client applications of *Fletcher*'s portlet proxy must be written specifically to invoke particular software resources. This is because, as explained above, the portlet proxy forwards the request onto the software resource bound to the portlet proxy. Therefore, *Fletcher*'s portlet proxy does not offer the advantages of the claimed "web services broker".

CLAIM 1

In making the rejection of Claim 1, the Office Action appears to equate the web services broker recited in Claim 1 with the portlet proxy of *Fletcher*. (*See Fletcher*, paragraph 46). However, in *Fletcher*, the portlet proxy does not broker requests between client applications and software resources. Rather, as explained above, it merely forwards such requests. Thus, *Fletcher* does not disclose or in any suggest a situation where the portlet proxy receives, from a

client application, a request for information from a particular software resource, where the client application does not have logic for directly interacting with the particular software resource and where the request is not compatible with the input data requirements of the particular software resource. Consequently, *Fletcher* does not disclose or in any way suggest the following limitations of Claim 1 which relate to characteristics of the claimed "client application" and the claimed "request" that are not disclosed in *Fletcher*:

> receiving at a web services broker, from a client application, a request for information from a particular web service, **the request having first data that cannot be used by said particular web service to service requests for said information**;
> ...
> wherein the client application is separate from the web services broker and **does not have logic for directly interacting with said particular web service**;

Further, to the extent that *Fletcher* teaches data transformation it does so only in the context of internal portlet-to-portlet messaging. *Fletcher*, paragraph 87. In contrast, Claim 1 features data transformation in the context of client application requests for information from web services that are separate from (i.e. external to) the web services broker. Thus, *Fletcher* does not teach or suggest the following limitations of Claim 1 which relate to receiving requests for external web services.

> wherein the particular web service serves as a source of said information, **is separate from the web services broker**, and has characteristics that are described in Web Service Description Language and are published in a Universal Description, Discovery, and Integration registry;
> **in response to receiving said request**, the web services broker **accessing transformation information** that specifies
> how to transform said first data associated with said request to second data that said particular web service can use to service requests for said requested information,
> how to invoke said particular web service in a manner required by said particular web service, to obtain said requested information from said particular web service, and
> how to transform a plurality of first data each from a respective client application of a plurality of client applications, to a plurality of second data each for a respective web service of a plurality of web services;
> **based on said transformation information**, the web services broker performing the steps of:

transforming said first data to said second data; and
invoking, in said manner required by said particular web service, said particular web service to obtain said requested information from said particular web service.

For these reasons, it is respectfully submitted that Claim 1 is allowable over *Fletcher*. Claim 49 has similar limitations, and is allowable for the same reasons.

CLAIM 17

In the prior response to the Office Action, it was explained why *Fletcher* does not teach or in any way suggest the following limitations of Claim 17:

receiving at a web services broker, from a client application, a request for information, wherein said request **includes an identification of a particular instance of said client application**
in response to receiving said request, **based on said identification of said particular instance of said client application, the web services broker accessing transformation information**;
wherein **said transformation information includes a mapping between said identification of said particular instance of said client application and an identification of said particular web service from which said particular instance wants said requested information**; (Emphasis added.) (In the prior response Claim 17 recited "source" where it now recites "client application".)

The Office Action asserts that these limitations are satisfied by *Fletcher* in paragraphs 10, 23, 46 and 72. In the Advisory Action mailed January 22, 2008 the assertion that these limitations are satisfied by *Fletcher* was maintained. Specifically, the Advisory Action asserts that "*Fletcher* discloses the data transformation based on identification of the requesting client application and that the transformation information includes a mapping between the identity of the client application and the identity of a particular web service per [0086]".

Paragraph 10 of *Fletcher* states:

> The core set of standards on which web services work is being built includes HTTP ("Hypertext Transfer Protocol"), SOAP ("Simple Object Access Protocol") and/or XML ("Extensible Markup Language") Protocol, WSDL ("Web Services Description Language"), and UDDI ("Universal Description, Discovery, and Integration"). HTTP is commonly used to exchange messages over TCP/IP ("Transmission Control Protocol/Internet Protocol") networks such as the Internet. SOAP is an XML-based protocol used to send messages for invoking methods in a distributed environment. XML Protocol is an evolving specification of the World Wide Web Consortium ("W3C") for an application-layer transfer protocol that will enable application-to-application messaging, and may converge with SOAP. WSDL is an XML format for describing distributed network services. UDDI is an XML-based registry technique with which businesses

may list their services and with which service requesters may find businesses providing particular services. (For more information on SOAP, refer to "Simple Object Access Protocol (SOAP) 1.1, W3C Note 8 May 2000", which is available on the Internet at http://www.w3.org/TR/2000/NOTE-SOAP-20000508. See http://www.w3.org/2000/xp for more information on XML Protocol and the creation of an XML Protocol standard. The WSDL specification is titled "Web Services Description Language (WSDL) 1.1, W3C Note 15 Mar. 2001", and may be found on the Internet at http://www.w3.org/TR/2001/NOTE-wsdl-20010315. For more information on UDDI, refer to the UDDI specification which is entitled "UDDI Version 2.0 API Specification, UDDI Open Draft Specification 8 Jun. 2001", and which can be found on the Internet at http://www.uddi.org/specification.html. HTTP is described in Request For Comments ("RFC") 2616 from the Internet Engineering Task Force, titled "Hypertext Transfer Protocol--HTTP/1.1" (June 1999).)

Paragraph 23 of *Fletcher* states:

The technique may further comprise using the populated system interface at run-time to manage the collection of software resources, and using the populated system may further comprise: retrieving, by a first retriever, the registered service description document for a selected service from the registry; binding, by the first retriever, to the populated system interface of the selected service using the retrieved service description document; programmatically notifying the selected service of an identifier of the first retriever; and establishing 2-way communications, by the selected service, using the identifier of the first retriever.

Paragraph 46 of *Fletcher* states:

A portal platform provides a number of services for the portlets it hosts, as described above. The present invention leverages portlets as a portal interface, and also builds upon the concept of a remote portlet interface (where this concept is extended as disclosed herein to apply to programmatic portlets), to enable access to software resources. Portlets functioning according to the present invention are also referred to herein as "web service intermediaries" or "web service proxies", or simply as "intermediaries" or "proxies". That is, a portlet may now act as an intermediary between an application or software resource requesting a particular service and a software resource providing that service. According to the present invention, the software resource performing a particular function may be statically bound to a web service proxy (for example, at development time), or a web service proxy may be bound to a software resource which is dynamically selected (for example, based upon criteria which are evaluated at run-time). In either case, the portlet proxy receives request messages and forwards them to the software resource to which it is bound; once the software resource has completed the requested function, it returns its response to the portlet proxy which then forwards the response to the requester.

Paragraph 72 of *Fletcher* states:

Returning now to the discussion of FIG. 7, once the composer completes the model of service interaction as a directed graph, at Block 710 the composer declares a service provider port type for the newly-aggregated service. In order to define this public interface, the composer may select operations to be exposed, or exported, within port types of the service of the composition, thereby identifying the public port type. The composer may select as many operations for exporting as necessary, and may also define new operations if desired. (For example, the composer might decide to define a new name for an operation, and provide a one-to-one mapping for that new name to the old name.) These exposed operations specify the means for invoking the new service. For example, with reference to the sample directed graph in FIG. 5, the composer would specify (at least) the operation "receivePurchaseOrder" corresponding to node 530 as a public interface. The composer may also provide a data mapping between an exposed operation and an internal service port type operation. For the sample directed graph in FIG. 5, having a public interface of "receivePurchaseOrder", the composer might specify (for example) a data mapping that converts data to a particular format required by this operation, such that an input parameter named "inputPurchaseOrder" adhering to a predefined type "purchaseOrder" would be available to the service upon invocation. Providing the data mapping may comprise identifying a canned transformation component, such as an integer-to-float transform; identifying a stylesheet which contains an appropriate transformation, such as an XSLT (Extensible Stylesheet Language Transformations) stylesheet; identifying customized transformation logic, which may for example convert one complex data type to another; and so forth. (Refer to the discussion of FIGS. 11A-11C, below, for more information about declaring the interface to a web service

> intermediary and potential web service software resources, and about providing transformation information.) The process of declaring the exposed port types (i.e. operations) results in the new service provider type for the aggregated composition. Thus, this new composition is itself a portlet proxy, or web service intermediary, that can be used as a building block in further compositions.

Paragraph 86 of *Fletcher* states:

> While plug links are known in the art, their use with portlet proxies is a novel aspect of the present invention. Note, however, that the data mapping support for plug links defined in the WSFL specification provides a relatively restricted form of data mapping. A <map> element is provided, as shown at 1410, for a <dataLink> element of a flow model, which qualifies how to map one operation's message to another. The plug link defined by WSFL is limited to specifying a source message, a single target message, a source message attribute, and a single target message attribute. Thus, if one web service returned the markup document 1500 shown in FIG. 15A, and needed to pass this information to an aggregated web service whose method signature was "submitName (Name: string)", the existing WSFL syntax allows specification of a mapping that would (for example) pass the value of the <firstName> element from document 1500 as the input parameter to the submitName method. An example <map> element 1520 which may be used within a plug link document to provide this mapping is shown in FIG. 15B. This is somewhat limiting in that operation attributes might not always map one-to-one, and transformations more complex than identifying an attribute for direct transfer might be necessary. For example, with reference to document 1500, it might be desirable to concatenate the values of the <firstName> and <lastName> elements for transfer to the next operation.

Obviously, these are very long excerpts that are alleged to show three important but relatively straight forward limitations, namely:

receiving at a web services broker, from a client application, a request for information, wherein said request **includes an identification of a particular instance of said client application**

in response to receiving said request, **based on said identification of said particular instance of said client application, the web services broker accessing transformation information**;

wherein **said transformation information includes a mapping between said identification of said particular instance of said client application and an identification of said particular web service from which said particular instance wants said requested information**;

Unfortunately, the Office Action and the Advisory Action is completely silent about what, within these excerpts, is supposed to satisfy these express limitations. Most significantly, the Office Action and Advisory Action do not give any indication as to what, within these excerpts, is supposed to be the claimed identification of a particular instance of the requesting client application included in the request.

For the purpose of argument, the Applicant assumes that the rejection is based on equating the "identifier of the first retriever" described in *Fletcher*, paragraph 23 with the

"identification of a particular instance of said client application" recited in Claim 1. If this is not the case, then the Applicant respectfully requests that the Examiner issue another office action that actually states what, within *Fletcher*, is supposed to be the claimed "identification of a particular instance of said client application".

It is respectfully submitted that "identifier of the first retriever" described in *Fletcher* does not satisfy the limitations, of Claim 1, that the Office Action alleges to be satisfied by *Fletcher*. Specifically, paragraph 23 of *Fletcher* discloses a technique for establishing 2-way communications between the portal platform and a portlet proxy. This technique is explained in detail by paragraphs 92 and 93 of *Fletcher*. (*see also Fletcher*, figure 17). In particular, to establish the 2-way communications, the portal platform provides an identifier to the portlet proxy through the portlet proxy's system interface. (*see Fletcher*, paragraph 93 stating "the (previously-registered) portal platform programmatically provides its URN to the portlet proxy (for example, by invoking a "setURN" method of the portlet proxy).") However, the portal platform of *Fletcher* cannot be a client application as featured in Claim 17 because the claimed client application is "separate from the web services broker". Further, nowhere in *Fletcher* is transformation information accessed based on the URN provided to the portlet proxy by the portal platform. In contrast, Claim 17 features "accessing transformation information" based on "said identification of said particular instance of said client application". Consequently, *Fletcher* does not satisfy:

> receiving at a web services broker, from a client application, a request for information, wherein said request **includes an identification of a particular instance of said client application**
> in response to receiving said request, **based on said identification of said particular instance of said client application, the web services broker accessing transformation information**;
> wherein **said transformation information includes a mapping between said identification of said particular instance of said client application and an identification of said particular web service from which said particular instance wants said requested information**;

For these reasons, it is respectfully submitted that Claim 17 is also allowable over *Fletcher*. Claim 50 has similar limitations, and is allowable for the same reasons.

DEPENDENT CLAIMS

All of the remaining claims depend on one of the claims discussed above. Therefore, each of these claims is allowable for those reasons. In addition, each of the dependent claims separately introduces limitations that render the claims allowable over the art. However, due to the fundamental differences already identified, and to expedite favorable resolution of this case, separate arguments are not provided for these claims.

CONCLUSION

For the reasons set forth above, it is respectfully submitted that all of the pending claims are now in condition for allowance. Therefore, the issuance of a formal Notice of Allowance is believed next in order, and that action is most earnestly solicited.

The Examiner is respectfully requested to contact the undersigned by telephone if it is believed that such contact would further the examination of the present application.

Please charge any shortages or credit any overages to Deposit Account No. 50-1302.

Respectfully submitted,

Hickman Palermo Truong & Becker LLP

Date: February 29, 2008

/AdamCStone#60531/
Adam Christopher Stone
Reg. No. 60,531

2055 Gateway Place, Suite 550
San Jose, California 95110-1089
Telephone No.: (408) 414-1080 ext. 231
Facsimile No.: (408) 414-1076